METHOD AND APPARATUS IN A TELECOMMUNICATIONS SYSTEM

TECHNICAL FIELD OF THE INVENTION

The present invention relates generally to methods for improving speech quality in e.g. IP-telephony systems. More particularly the present invention relates to a method for reducing audio artefacts due to overrun or underrun in a playout buffer.

The invention also relates to an arrangement for carrying out the method.

DESCRIPTION OF RELATED ART

When sampling frequencies, in e.g. a speech coding system, are not controlled, underrun or overrun might occur in the playout buffer, which is a buffer storing speech samples for later playout. Underrun means that the playout buffer will run into starvation, i.e. it will no longer have any samples to play on the output. Overrun means that the playout buffer will be filled with samples and that following samples cannot be buffered and consequently will be lost. Underrun is probably more common than overrun since the size of the playout buffer can increase until there is no memory left, while it only can decrease until there are no samples left.

Currently, most systems do not deal with the problem that the sampling frequency might differ considerably between the sending and the receiving side. One possible solution proposed in, EP-0680033 A2, works on pitch periods. Adding or removing pitch periods in the speech signal achieves a different duration of a speech segment without affecting other speech characteristics than speed. This proposed solution might be used as an indirect sample rate conversion method.

Another solution uses the beginning of talkspurts as an indication to reset the playout buffer to a specified level. The distance, in number of samples, between two consecutive talkspurts is increased if the receiving side is playing faster than the sending side and decreased if the receiving side is playing slower than the sending side. In IP-telephony solutions, using the IP/UDP/RTP-protocols (Internet Protocol/User Datagram Protocol/Real Time Protocol); the marker flag in the RTP header is used to identify the beginning of a talkspurt. At the beginning of a talkspurt the playout buffer is set to a suitable size.

The solution according to EP-0680033 A2, where pitch periods are removed or inserted, assumes a fixed conversion factor between the receiving and transmitting side. Therefore it cannot be used in dynamical systems, i.e. where the sampling frequencies varies. Further, it does not solve the problem with underrun or overrun situations, but is instead focused on changing the playback rate of a speech signal stored in compressed form for playback later and at another speed compared to when it was stored.

Using the method of resetting the playout buffer to a certain size causes problems if there are very long talkspurts, e.g. broadcast from one speaker to several listeners. Since the length of a talkspurt is not defined in the beginning of the talkspurt the size to reset to might be either too small or too large. If it is too small, underrun will occur and if it is too large, unnecessary delay is introduced, thus the problem persists.

The general problem with the currently known approaches is that they are static and inflexible. As a conclusion dynamic solutions are required.

SUMMARY OF THE INVENTION

The present invention deals with the problem of improving speech quality in systems where the sampling rate at a transmitting terminal differs from the playout rate of a receiving buffer at a receiving terminal. This is often the case in e.g. IP-telephony.

sub a7 When sampling frequencies are not controlled, underrun or overrun might occur in the playout buffer at the receiving side, which causes audible artefacts in the speech signal. To avoid said overrun or underrun there is a need for dynamically keeping the playout buffer to an average size, i.e. controlling the fullness of the playout buffer.

sub a8 One object of the present invention is thus to provide a method for reducing audio artefacts in a speech signal due to overrun or underrun in the playout buffer.

sub a9 Another object of the invention is to dynamically control the fullness of the playout buffer as not to introduce extra delay.

The above mentioned objects are achieved by means of dynamic sample rate conversion of speech frames, i.e. converting speech frames comprising N samples to instead comprise either N+1 or N-1 samples. More specifically the invention works on an LPC-residual of the speech frame and by adding or removing a sample in the LPC-residual, a sample rate conversion will be achieved. The LPC-residual is the output from an LPC-filter, which removes the short-term correlation from the speech signal. The LPC-filter is a linear predictive coding filter where each sample is predicted as a linear combination of previous samples.

sub a10

sub a11 By using the proposed sample rate conversion method, the playout buffer, of e.g. an IP-telephony terminal, can be continuously controlled with only small audio artefacts. Since the method works on a sample-by-sample basis the playout buffer

can be kept to a minimum and hence no extra delay is introduced. The solution also has very low complexity, especially when the LPC-residual already is available, which is the case in e.g. a speech decoder.

The term "comprises/comprising" when used in this specification is taken to specify the presence of stated features, integers, steps or components but does not preclude the presence or addition of one or more other features, integers, steps, components or groups thereof.

Although the invention has been summarised above, the method and arrangement according to the appended independent claims 1 and 23 define the scope of the invention. Various embodiments are further defined in the dependent claims 2-12 and 24-44.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 shows a transmitter and a receiver to which the method of the invention can be applied.

Figure 2 shows a speech signal in the time domain.

Figure 3 shows an LPC-residual of a speech signal in the time domain.

Figure 4 illustrates four modules of the sample rate conversion method according to the invention.

Figure 5A shows an analysis-by-synthesis speech encoder with LTP-filter.

Figure 5B shows an analysis-by-synthesis speech encoder with adaptive codebook.

Figures 5C-5F show different implementations of the LPC-residual extraction depending on the realisation of the speech encoder.

Figures 5G-5J show four ways of placing the sample rate conversion within the feed back loop of the speech decoder.

Figure 6 illustrates how to use information about pitch pulses to find samples with low energy.

Figure 7 illustrates LPC-history extension.

Figure 8 illustrates copying of the history of the LPC residual.

DETAILED DESCRIPTION

The present invention describes, referring to figure 1, a method for improving speech quality in a communication system comprising a first terminal unit TRX1 transmitting speech signals having a first sample frequency F_1 and a second terminal unit TRX2 receiving said speech signals, buffering them in a playout buffer 100 with said first frequency F_1 and playing out from said playout buffer with a second frequency F_2. When the buffering frequency F_1 is larger than the playout frequency F_2 the playout buffer 100 will eventually be filled with samples and subsequent samples will have to be discarded. When the buffering frequency F_1 is lower than the playout frequency F_2 the playout buffer will run into starvation, i.e. it will no longer have any samples to play on the output. These two problems are called overrun and underrun respectively, and causes audible artefacts like popping and clicking sounds in the speech signal.

The above problems with underrun and overrun are solved by using dynamic sample rate conversion based on modifying the LPC-residual of the speech signal and will be further described with reference to figures 2-8.

Figure 2 shows a typical segment of a speech signal in the time domain. This speech signal shows a short-term correlation, which corresponds to the vocal tract and a long-term correlation, which corresponds to the vocal cords. The short-term correlation can be predicted by using an LPC-filter and the long-term correlation can be predicted by using an LTP-filter. LPC means linear predictive coding and LTP means long

term prediction. Linear in this case implies that the prediction is a linear combination of previous samples of the speech signal.

The LPC-filter is usually denoted:

$$H(z)=\frac{1}{A(z)}=\frac{1}{1-\sum_{i=1}^{n}a_i z^{-i}}$$

By feeding a speech frame through the LPC-filter, $H(z)$, the LPC-residual is found. The LPC-residual, shown in figure 3, contains pitch pulses P generated by the vocal cords. The distance L between two pitch pulses P is called lag. The pitch pulses P are also predictable, and since they represent the long-term correlation of the speech signal they are predicted through an LTP-filter given by the distance L between the pitch pulses P and the gain b of a pitch pulse P. The LTP-filter is usually denoted:

$$F(z)=b\cdot z^{-L}$$

When the LPC-residual is fed through the inverse of the LTP-filter $F(z)$ an LTP-residual is created. In the LTP-residual the long-term correlation in the LPC-residual is removed, giving the LTP-residual a noise-like appearance.

The solution according to the invention modifies the LPC-residual, shown in figure 3, on a sample-by-sample basis. That is, an LPC-residual block comprising N samples is converted to an LPC-residual block comprising either N+1 or N-1 samples. The LPC-residual contains less information and less energy compared to the speech signal but the pitch pulses P are still easy to locate. When modifying the LPC-residual, samples being close to a pitch pulse P should be avoided, because these samples

contain more information and thus have a large influence on the speech synthesis. The LTP-residual is not as suitable as the LPC-residual to use for modification since the pitch pulse positions P are no longer available. As a conclusion, the LPC-residual is better suited for modification both compared to the speech signal and the LTP-residual, since the pitch pulses P are easily located in the LPC-residual.

The proposed sample rate conversion consists of four modules, shown in figure 4:

1) A Sample Rate Controller (SRC) module 400 that calculates whether a sample should be added or removed;

2) LPC-Residual Extraction (LRE) modules 410 are used to obtain the LPC-residual r_{LPC};

3) Sample Rate Conversion Methods (RCM) modules 420 find the position where to add or remove samples and how to perform the insertion and deletion, i.e. converting the LPC residual block r_{LPC} comprising N samples to a modified LPC-residual block r'_{LPC} comprising N+1 or N-1 samples; and

4) A Speech Synthesiser Module (SSM) 430 to reproduce the speech.

The idea behind the invention is that it is possible to change the playout rate of the playout buffer 440 by removing or adding samples in the LPC-residual r_{LPC}.

The SRC module 400 decides whether samples should be added or removed in the LPC residual r_{LPC}. This is done on the basis of at least one of the following parameters; the sampling frequencies of the sending TRX1 and receiving terminal units

TRX2, information about the speech signal e.g. a voice activity detector signal, status of the playout buffer or an indicator of the beginning of a talkspurt. These inputs are named SRC Inputs in the figure. On the basis of a function of one or several of these parameters the SRC 400 forms a decision on when to insert or remove a sample in the LPC residual r_{LPC} and optionally which RCM 420 to use. Since digital processing of speech signals usually is made on a frame-by-frame basis, the decision on when to remove or add samples basically is to decide within which LPC-residual r_{LPC} frame the RCM 420 shall insert or remove a sample.

There are basically three methods of obtaining the LPC-residual r_{LPC} that is needed as input to the RCM's 420. The methods depend on the implementation of the speech encoder and will be described with reference to figures 5A-5F. The LRE solution also directly influences the SSM solution, which will become apparent below.

In figure 5A is an analysis-by-synthesis speech encoder 500 with LTP-filter 540 shown. This is a hybrid encoder where the vocal tract is described with an LPC-filter 550 and the vocal cords is described with an LTP-filter 540, while the LTP-residual $\hat{r}_{LTP}(n)$ is waveform-compared with a set of more or less stochastic codebook vectors from the fixed codebook 530. The input signal S is divided into frames 510 with a typical length of 10-30 ms. For each frame an LPC-filter 550 is calculated through an LPC-analysis 520 and the LPC-filter 550 is included in a closed loop to find the parameters of the LTP-filter 540. The speech decoder 580 is included in the encoder and consists of the fixed codebook 530 which output $\hat{r}_{LTP}(n)$ is connected to the LTP-filter 540 which output $\hat{r}_{LPC}(n)$ is connected to the LPC-filter 550 generating an estimate $\hat{s}(n)$ of the original speech signal $s(n)$. Each estimated signal $\hat{s}(n)$ is compared with the original speech signal $s(n)$ and a difference signal $e(n)$ is

calculated. The difference signal $e(n)$ is then weighted 560 to calculate a perceptual weighted error measure $e_w(n)$. The set of parameters that gives the least perceptual weighted error measure $e_w(n)$ is transmitted to the receiving side 570.

As can be seen in figure 5C the LPC-residual $\hat{r}_{LPC}(n)$ is the output from the LTP-filter 540. The SRC/RCM modules 545 can thus be connected directly to that output and integrated into the speech encoder. The LRE consists of the fixed codebook 530 and the long-term predictor 540 and the SSM consists of an LPC-filter 550, thus the LRE-module and the SSM-module are natural parts of the speech decoder.

If the speech encoder, on the other hand, is an analysis-by-synthesis speech encoder where the LTP-filter 540 is exchanged to an adaptive codebook 590 as shown in figure 5B, the LPC-residual $\hat{r}_{LPC}(n)$ is the output from the sum of the adaptive and the fixed codebook 590 and 530. All other elements have the same function as in figure 5A showing the analysis-by-synthesis speech encoder with LTP-filter 500. As can be seen in figure 5D the LPC residual $\hat{r}_{LPC}(n)$ is the sum of the output from the adaptive and fixed codebook 590 and 530. The SRC/RCM modules 545 can thus again be connected directly to that output and integrated into the speech encoder as shown in figure 5D. The LRE consists of the adaptive and the fixed codebook 590 and 530 and the SSM consists of an LPC-filter 550, thus the LRE module and the SSM module are again natural parts of the speech decoder.

When the speech encoder has some sort of backward adaptation, it is not feasible to make alterations in the LPC-residual since this would affect the adaptation process in a detrimental way. In figure 5E is shown how in these cases the parameters $\hat{s}(n)$ from the LPC-filter 550 could be fed to an inverse LPC-filter 525 placed after the speech decoder. After the sample

rate conversion has been made in the SRC/RCM modules 545 an LPC-filtering 550 is performed to reproduce the speech signal. The LRE module consists of the inverse LPC-filter 525 and the SSM module consists of the LPC-filter 550.

In figure 5F is shown how it is possible to produce an LPC residual $\hat{r}_{LPC}(n)$ through a full LPC analysis. The output $\hat{s}(n)$ from the speech decoder is fed to both an LPC analysis block 520 and an LPC-inverse filter 525. After the sample rate conversion has been made in the SRC/RCM modules 545, an LPC filtering 550 is performed to reproduce the speech signal. The LRE consists in this case of the LPC analysis 520 respective the LPC inverse filter 525 and the SSM module consists of the LPC filter 550. Performing an LPC analysis is considered to be well known to a person skilled in the art and is therefore not discussed any further.

Referring again to figure 4, assume that the SRC-module 400 has decided that a sample should be added or removed in the LPC residual r_{LPC} and that the LRE module 410 has produced an LPC residual r_{LPC}. The RCM-module 420 then has to find the exact position in the LPC-residual r_{LPC} where to add or remove a sample and performing the adding respective removing. There are four different methods for the RCM-module 420 to find the insertion or deletion point.

The first and most primitive method arbitrarily removes or adds a sample whenever this becomes necessary. If the sample rate difference between the terminals is small this will only lead to minor artefacts since the adding or removing is performed very seldom.

By inserting or removing samples at positions where the energy in the LPC-residual is low the synthesis will be less affected. This is due to the fact that segments close to pitch pulses

will then be avoided. To find these segments of low energy either a sliding window method or a simpler block energy analysis can be used.

The second method, called the sliding window energy method, calculates a weighted energy value for each sample in the LPC-residual. This is done by multiplying k samples surrounding a sample with a window function of size k (k<<N), where N equals the number of samples in the LPC-residual. Each sample is then squared and the sum of the resulting k values is calculated. The window is shifted one position and the procedure is repeated. The position where to insert or remove samples is given by the sample with the lowest weighted energy value.

The third method, block energy analysis, is a simpler solution for finding the insertion or deletion point. The LPC-residual is simply divided into blocks of equal length and an arbitrary sample is removed or added in the block with the lowest energy.

sub-a31

The fourth method, as illustrated in figure 6, uses knowledge about the position P of a pitch pulse, and the lag L between two pitch pulses. With knowledge about that, it is possible to calculate a position P' having low energy and where it is therefore appropriate to add or remove a sample. The new position P' can be expressed as $P' = P + k \cdot L$ where the constant k is selected so that P' is selected to be somewhere in the middle between two pitch pulses, thus avoiding positions with high energy. A typical value of k is in the range of 0.5 to 0.8.

sub-a32

When the RCM-module 420 has calculated the position where to add or remove a sample it must be determined how to perform the insertion or deletion. There are three methods of performing such insertion or deletion depending on the type of LRE-module used.

sub a33 In the first method either zeros are added or samples with small amplitudes are removed. This method can be used for all LRE solution described above, see figures 5C-5F. Notice that in figures 5C and 5D the SRC/RCM-modules are placed before the synthesis filter SSM, but after the feed back of the LPC residual to the LTP-filter 540 respective the adaptive codebook 590.

sub a34 In the second method insertion is carried out by adding zeros and interpolating surrounding samples. Deletion is performed by removing samples and preferably smoothing surrounding samples. This method can also be used for all of the LRE solutions described above, see figures 5C-5F. Notice that in figure 5C and 5D the SRC/RCM-modules are placed before the synthesis filter SSM, but after the feed back of the LPC residual to the LTP-filter 540 respective the adaptive codebook 590.

sub a35 In the third method the SRC/RCM-modules 545 are placed within the feedback loop of the speech decoder, see figures 5G-5J, instead of after the feedback loop as in the previous methods. Placing the SRC/RCM-modules within the feedback loop uses real LPC residual samples for the sample rate conversion, by changing the number of components in the LPC-residual. The implementation differs depending on whether it is an analysis-by-synthesis speech encoder with LTP filter shown in figure 5A or an analysis-by-synthesis speech encoder with adaptive codebook shown in figure 5B, that is used.

sub a36 For the speech decoder with LTP filter, see figure 5A, the SRC/RCM-modules 545 can be placed within the feedback loop in two different ways, either within the LTP feedback loop as shown in figure 5G or in the output from the fixed codebook 530 as shown in figure 5H. For the speech decoder with adaptive codebook, see figure 5B, the SRC/RCM can also be placed in two different ways, i.e. either before, figure 5J, or after, figure

sub a36 cont

5I, the summation of the outputs from the adaptive and the fixed codebook.

The alterations on the LPC residual consists of removing or adding samples just as before but since the SRC/RCM-modules 545 are placed within the LTP feedback loop some modifications must be done. The extending or shortening of a segment can be done in three ways either at the respective ends of the segment or somewhere in the middle of the segment. Figure 7 shows the case where the LPC residual is extended by copying two overlapping segments, segment 1 and segment 2, from the history of the LPC residual to create the longer LPC residual. The normal case when no insertion or deletion is needed would be to copy N samples. Shortening the LPC residual is achieved by copying two segments that has a gap between them instead of being overlapped. As before, it is important that a pitch pulse is not doubled or removed since this would introduce perceptual artefacts. Hence, an analysis should be performed in order to evaluate where to add or remove segments. This analysis is preferably made by using the same methods as discussed above regarding how to find the position where to add or remove a sample in the RCM-module.

sub a37

For all implementations except when the SRC/RCM-modules 545 are placed between the fixed codebook 530 and the LTP filter 540 the history of the LPC residual also has to be modified. The lag L will be increased or decreased for the specific part of the history where a sample is inserted or deleted. Thus the starting position of the segment that will be copied from the history of the LPC residual, Pointer 1 or Pointer 2 in figure 8, needs modification. If the segment to copy is newer, i.e. the case of Pointer 1, there is no need to modify the starting position. If, however, the segment to copy is older, i.e. the case of Pointer 2, then the pointer should be increased or decreased depending on if a sample is inserted or deleted. This

has to be managed for subsequent sub-frames and frames as long as the modification is within the history of the LPC residual.

When the SRC/RCM-modules are placed before the summation of the outputs from the adaptive and the fixed codebook as shown in figure 5J the length of the fixed codebook also needs to be changed. This is done by adding a sample, preferably a zero sample, in the output from the fixed codebook or removing one of the components. The insertion and deletion in the fixed codebook should be synchronised with the insertion and deletion in the adaptive codebook.

The invention being thus described, it will be obvious that the same may be varied in many ways. Such variations are not to be regarded as a departure from the scope of the invention, and all such modifications as would be obvious to a person skilled in the art are intended to be included within the scope of the following claims.